Анализ сетевых данных и обнаружение аномалий с использованием методов машинного обучения и экстремального анализа

Студент: Гущин Станислав Алексеевич
Научный руководитель: Горшенин Андрей Константинович

Содержание

1 Введение

Целью данной работы является анализ большого объёма сетевых данных (более 1.3 миллиона пакетов) с целью выявления аномальных потоков и нетипичного поведения в трафике. Для решения задачи использовались методы машинного обучения (VAE и кластеризация KMeans), а также статистический анализ экстремальных значений POT. Анализ проводился в среде Jupyter Notebook на языке Python с использованием библиотек pandas, matplotlib, seaborn, scikit-learn, torch и др.

2 Описание данных

Входной набор данных представляет собой CSV-файл, содержащий следующие столбцы:

- **No.** — номер пакета в последовательности;
- **Time** — время захвата пакета;
- **Source** — IP-адрес источника;
- **Destination** — IP-адрес получателя;
- **Protocol** — используемый сетевой протокол (TCP, UDP, DNS, WireGuard и т.д.);
- **Length** — размер пакета в байтах;
- **Info** — дополнительная информация о пакете (например, флаги TCP).

Рисунок 1. Первые и последние строки датафрейма с данными:

	No.	Time	Source	Destination	Protocol	Length	Info
0	1	0.000000	185.7.214.20	158.250.17.140	TCP	60	53396 > 6547 [SYN] Seq=0 Win=1025 Len=0 MSS=...
1	2	0.012582	157.15.202.15	158.250.17.181	TCP	60	55501 > 30553 [SYN] Seq=0 Win=1024 Len=0
2	3	0.020753	83.222.190.246	158.250.17.240	TCP	60	56806 > 11944 [SYN] Seq=0 Win=1024 Len=0
3	4	0.023914	83.222.190.246	158.250.17.184	TCP	60	56806 > 14826 [SYN] Seq=0 Win=1024 Len=0
4	5	0.045710	213.87.130.79	188.44.42.233	WireGuard	122	Transport Data, receiver=0x7B4BBC8A, counter=3...
...	...	...	...	...	...	...	...
1375908	1375909	3599.923643	172.21.253.120	172.20.253.42	TLSv1.2	345	Application Data[Packet size limited during ca...
1375909	1375910	3599.923846	172.20.253.42	172.21.253.120	TLSv1.2	473	Application Data[Packet size limited during ca...
1375910	1375911	3599.924306	210.89.45.84	158.250.17.30	TCP	60	55533 > 35293 [SYN] Seq=0 Win=1024 Len=0
1375911	1375912	3599.928318	45.142.212.229	158.250.17.159	TCP	60	42562 > 2064 [SYN] Seq=0 Win=1025 Len=0 MSS=...
1375912	1375913	3599.928528	185.7.214.20	158.250.17.63	TCP	60	53380 > 9254 [SYN] Seq=0 Win=1025 Len=0 MSS=...

1375913 rows × 7 columns

Рис. 1: Просмотр загруженных сетевых пакетов

В наборе содержится **1 375 913 строк и 7 столбцов**. Это представляет собой большой объём трафика, типичный для сетевого мониторинга.

3 Предварительная визуализация данных

Для первичного понимания структуры трафика были построены графики зависимости времени от источников и назначения пакетов. Также выполнен групповой анализ по длине пакетов:

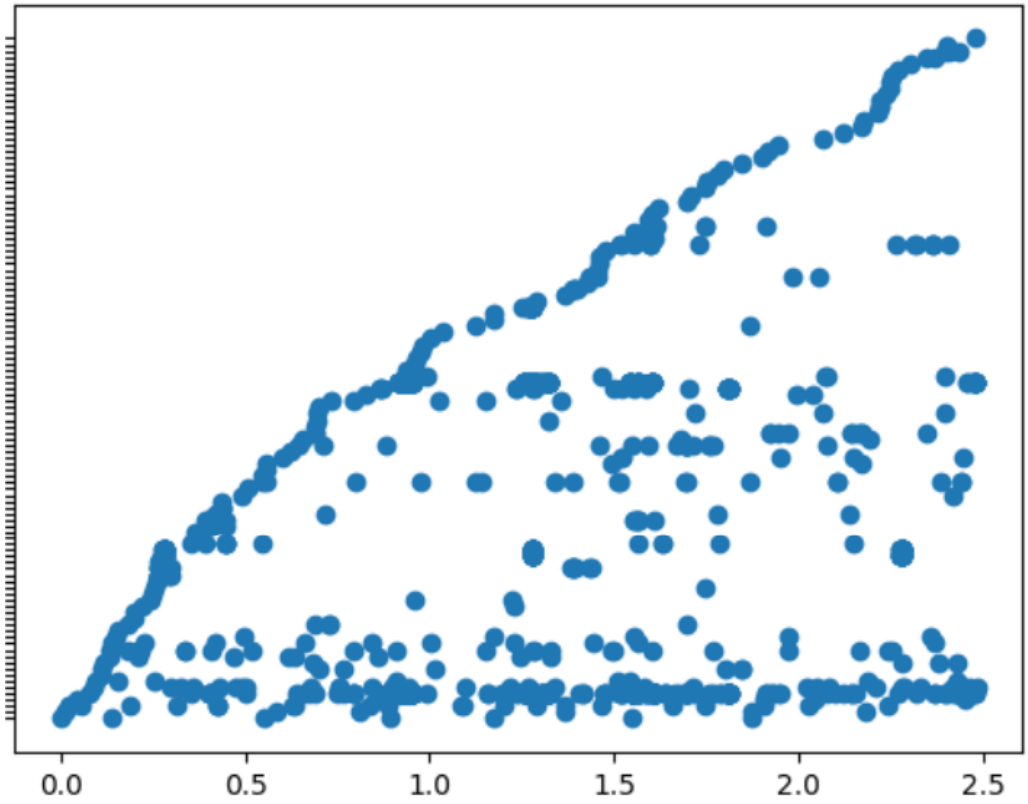

Рис. 2: Точечная диаграмма: Source (y) vs Time (x) (первые 1000 записей)

График показывает, что активность IP-адресов распределена неравномерно во времени. Для более точного анализа данных было выполнено агрегирование по потокам.

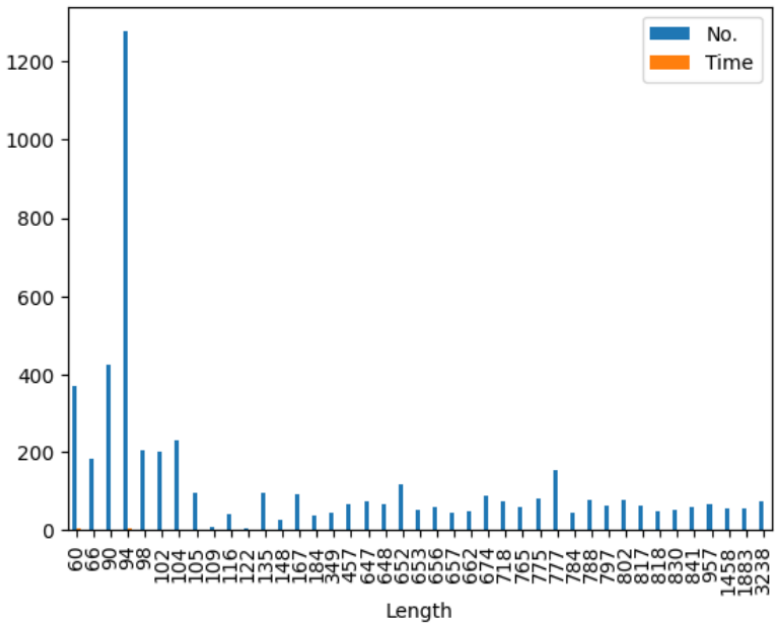

Рис. 3: Группировка по длине пакета: частоты встречаемости

Из графика видно, что преобладают пакеты длиной 94 байт. Это характерно для SYN-пакетов TCP, инициирующих соединение.

4 Агрегирование потоков

Данные были преобразованы в потоковую форму — сгруппированы по комбинациям IP-адресов, протоколов и других признаков. Получены метрики:

- общее число пакетов в потоке (packet count),
- продолжительность потока (duration),
- средний, минимальный, максимальный и стандартное отклонение размера пакетов.

Итоговая таблица содержит **754 255 потоков**.

	Source	Destination	Protocol	Flow_ID	packet_count	duration	mean_packet_size	std_packet_size	min_packet_size	max_packet_size
0	0.0.0.0	224.0.0.1	IGMPv2	16	1	0.0	60.0	0.0	60	60
1	0.0.0.0	224.0.0.1	IGMPv2	253	1	0.0	60.0	0.0	60	60
2	0.0.0.0	224.0.0.1	IGMPv2	19566	1	0.0	60.0	0.0	60	60
3	0.0.0.0	224.0.0.1	IGMPv2	54078	1	0.0	60.0	0.0	60	60
4	0.0.0.0	224.0.0.1	IGMPv2	93473	1	0.0	60.0	0.0	60	60
...	...	...	...	...	...	...	...	...	...	...
754250	fe80::ff5c:7475:dd70:9bcb	ff02::fb	MDNS	45028	1	0.0	202.0	0.0	202	202
754251	fe80::ff5c:7475:dd70:9bcb	ff02::fb	MDNS	50702	1	0.0	260.0	0.0	260	260
754252	fe80::ff5c:7475:dd70:9bcb	ff02::fb	MDNS	50751	1	0.0	102.0	0.0	102	102
754253	fe80::ff5c:7475:dd70:9bcb	ff02::fb	MDNS	50812	1	0.0	102.0	0.0	102	102
754254	fe80::ff5c:7475:dd70:9bcb	ff02::fb	MDNS	50902	1	0.0	102.0	0.0	102	102

754255 rows × 10 columns

Рис. 4: Пример агрегированных потоков

—

5 Архитектура вариационного автоэнкодера (VAE)

Для решения задачи выделения скрытых признаков и обнаружения аномалий была реализована и обучена модель вариационного автоэнкодера (VAE).

Структура модели

Модель состоит из двух основных частей: энкодера и декодера, объединённых в один класс ‘VAE‘.

- **Входной слой:** размерностью D, соответствующей числу признаков аггрегированного потока.
- **Скрытый слой энкодера:** линейное преобразование $\mathbb{R}^D \to \mathbb{R}^{256}$ с активацией ReLU.
- **Выход энкодера:** два линейных слоя, выдающие параметры латентного распределения:
  - μ — вектор средних значений;
  - $\log \sigma^2$ — логарифм дисперсий (для обеспечения положительности).
- **Reparameterization trick:** используется для генерации латентной переменной z по формуле:
$$z = \mu + \epsilon \cdot \exp(0.5 \cdot \log \sigma^2), \quad \epsilon \sim \mathcal{N}(0, I)$$
- **Декодер:** последовательность слоёв:
  - $\mathbb{R}^2 \to \mathbb{R}^{256}$ — скрытый слой декодера с ReLU;
  - $\mathbb{R}^{256} \to \mathbb{R}^D$ — выходной слой, восстанавливающий исходные признаки.

Функция потерь

Общая функция потерь состоит из двух компонентов:

- **Ошибка реконструкции** — среднеквадратичная ошибка между исходными и восстановленными признаками:
$$\mathcal{L}_{\text{recon}} = \text{MSE}(x, \hat{x})$$
- **Дивергенция Кульбака–Лейблера** между аппроксимирующим распределением $q(z|x)$ и стандартным нормальным распределением $p(z)$:
$$\mathcal{L}_{\text{KL}} = -\frac{1}{2} \sum \left(1 + \log \sigma^2 - \mu^2 - \sigma^2\right)$$

Итоговая функция потерь:
$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{KL}}$$

Применение и обучение

Модель обучалась на нормализованных потоках с размерностью признакового пространства $D = 6$. Размерность латентного пространства выбрана равной 2 — это позволяет не только выделить ключевые вариации в данных, но и визуализировать латентные кластеры.

Обучение проводилось в течение 20 эпох с использованием оптимизатора Adam (скорость обучения 10^{-4}). Для оценки качества модели использовалась ошибка реконструкции как на тренировочной, так и на тестовой выборках.

Полученные латентные представления использовались в дальнейшем для кластеризации с помощью KMeans, что позволило выделить группы, потенциально содержащие аномалии.

6 Обучение вариационного автоэнкодера (VAE)

Для понижения размерности и выявления скрытых признаков был применён вариационный автоэнкодер (VAE). Он обучается восстанавливать входные потоки и при этом извлекает скрытое (латентное) представление потока.

Модель обучалась в течение 15 эпох. Ниже представлены значения функции потерь (loss):

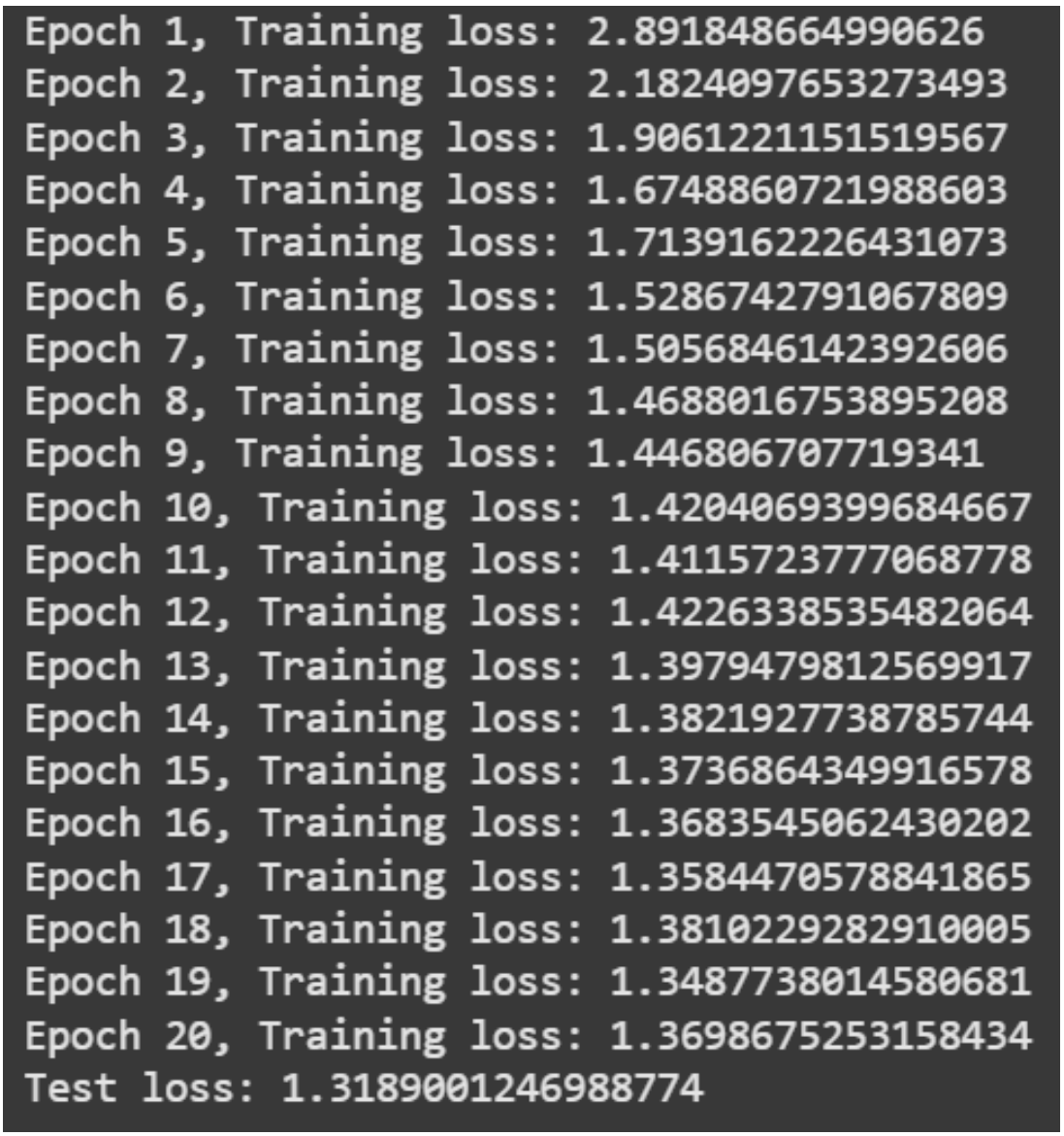

Рис. 5: Процесс обучения VAE

7 Кластеризация латентных признаков (KMeans)

После обучения VAE были извлечены латентные вектора mu и выполнена кластеризация методом KMeans с числом кластеров $k = 3$.

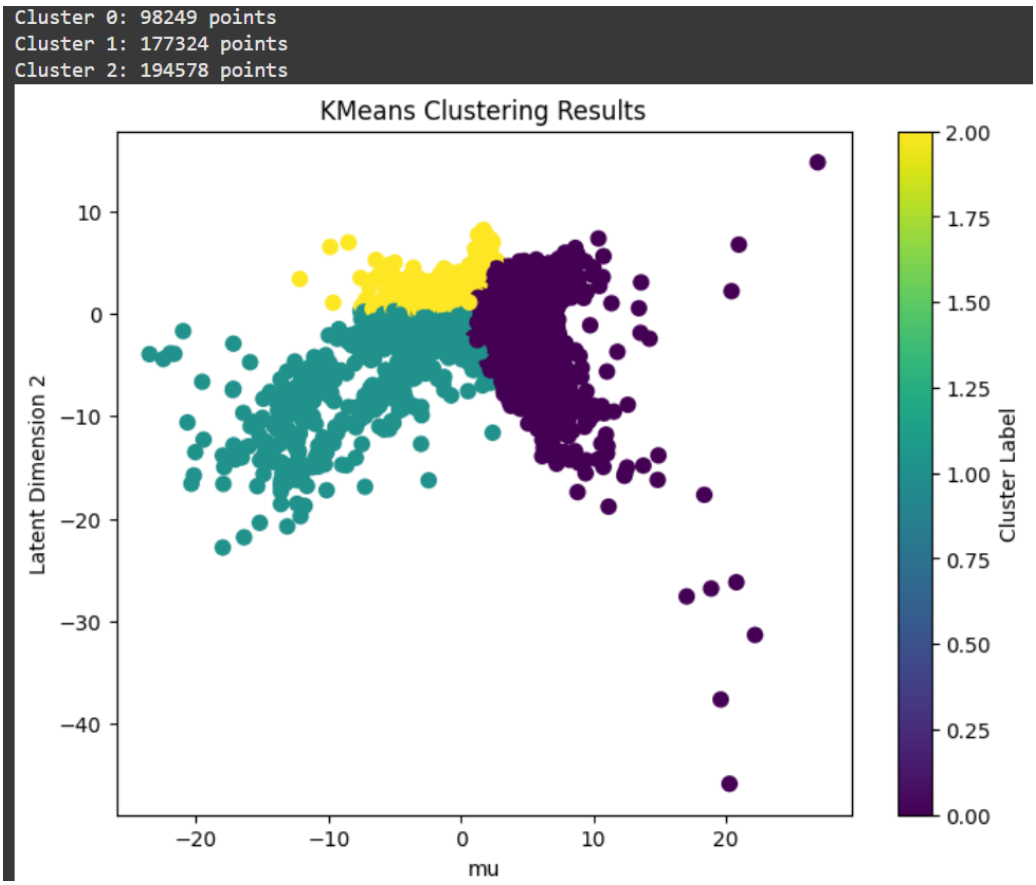

Рис. 6: Результат кластеризации KMeans в латентном пространстве

Выделены 3 кластера, каждый из которых соответствует различным типам поведения трафика. Наибольшее число потоков наблюдается в кластере 2 (более 190 тыс. записей).

8 Обнаружение аномалий через ошибку реконструкции

Вариационный автоэнкодер позволяет не только выделить латентные признаки, но и измерить точность реконструкции потока. Чем больше ошибка реконструкции — тем более вероятна аномалия.

Был выбран порог ошибки реконструкции, превышение которого интерпретировалось как аномалия:

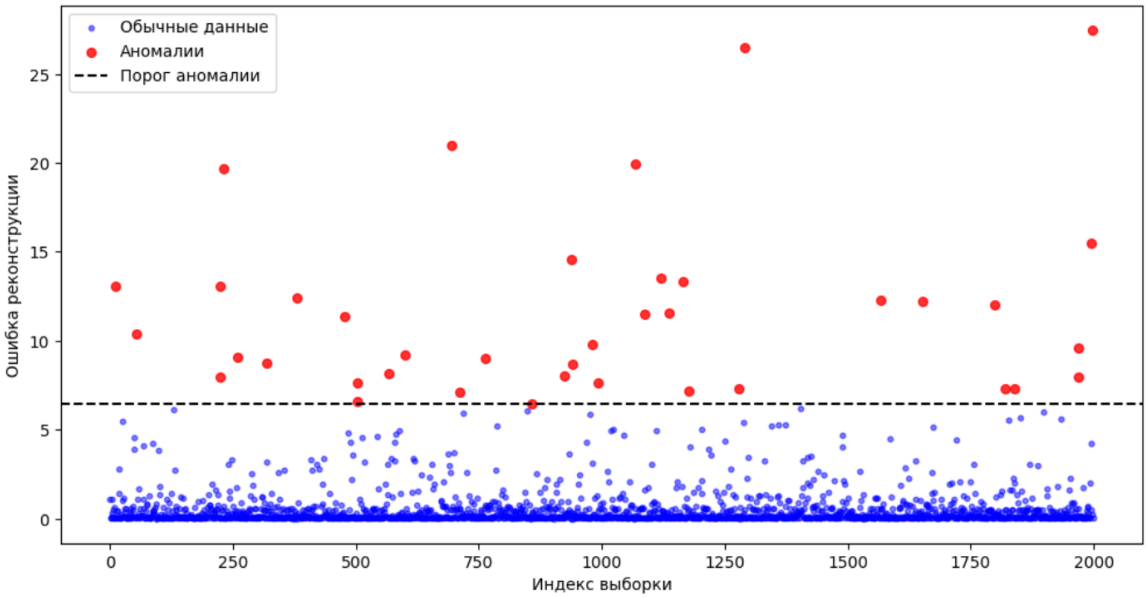

Рис. 7: Разделение на обычные и аномальные данные по ошибке реконструкции. В качестве порога подбирались $\mu + 3\sigma$ - среднее от всех ошибок реконструкции, стандартное отклонение при реконструкции, соответственно.

9 Сравнение характеристик аномалий

20 обычных данных:

	Source	Destination	Protocol	Flow_ID	packet_count	duration
58	0.0.0.0	224.0.0.1	IGMPv3	50572	1	0.0
59	0.0.0.0	224.0.0.1	IGMPv3	52330	1	0.0
60	0.0.0.0	224.0.0.1	IGMPv3	53013	1	0.0
61	0.0.0.0	224.0.0.1	IGMPv3	54010	1	0.0
62	0.0.0.0	224.0.0.1	IGMPv3	55278	1	0.0
63	0.0.0.0	224.0.0.1	IGMPv3	56953	1	0.0
64	0.0.0.0	224.0.0.1	IGMPv3	57792	1	0.0
65	0.0.0.0	224.0.0.1	IGMPv3	58613	1	0.0
66	0.0.0.0	224.0.0.1	IGMPv3	63202	1	0.0
67	0.0.0.0	224.0.0.1	IGMPv3	68923	1	0.0
68	0.0.0.0	224.0.0.1	IGMPv3	71536	1	0.0
69	0.0.0.0	224.0.0.1	IGMPv3	71737	1	0.0
70	0.0.0.0	224.0.0.1	IGMPv3	73729	1	0.0
71	0.0.0.0	224.0.0.1	IGMPv3	74311	1	0.0
72	0.0.0.0	224.0.0.1	IGMPv3	74880	1	0.0
73	0.0.0.0	224.0.0.1	IGMPv3	78534	1	0.0
74	0.0.0.0	224.0.0.1	IGMPv3	79046	1	0.0
75	0.0.0.0	224.0.0.1	IGMPv3	80215	1	0.0
76	0.0.0.0	224.0.0.1	IGMPv3	80273	1	0.0
77	0.0.0.0	224.0.0.1	IGMPv3	80799	1	0.0

	mean_packet_size	std_packet_size	min_packet_size	max_packet_size
58	60.0	0.0	60	60
59	60.0	0.0	60	60
60	60.0	0.0	60	60
61	60.0	0.0	60	60
62	60.0	0.0	60	60
63	60.0	0.0	60	60
64	60.0	0.0	60	60
65	60.0	0.0	60	60
66	60.0	0.0	60	60
67	60.0	0.0	60	60
68	60.0	0.0	60	60
69	60.0	0.0	60	60
70	60.0	0.0	60	60
71	60.0	0.0	60	60
72	60.0	0.0	60	60
73	60.0	0.0	60	60
74	60.0	0.0	60	60
75	60.0	0.0	60	60
76	60.0	0.0	60	60
77	60.0	0.0	60	60

20 аномальных данных:

	Source	Destination	Protocol	Flow_ID	packet_count	duration
319	1.0.0.1	188.44.42.233	DNS	2928	1	0.000000
380	1.0.0.1	188.44.42.233	DNS	8133	2	0.003174
477	1.0.0.1	188.44.42.233	DNS	16622	2	0.021272
502	1.0.0.1	188.44.42.233	DNS	17233	1	0.000000
503	1.0.0.1	188.44.42.233	DNS	17276	2	0.003610
567	1.0.0.1	188.44.42.233	DNS	18094	1	0.000000
599	1.0.0.1	188.44.42.233	DNS	18937	1	0.000000
695	1.0.0.1	188.44.42.233	DNS	23825	1	0.000000
711	1.0.0.1	188.44.42.233	DNS	25221	2	0.001299
763	1.0.0.1	188.44.42.233	DNS	28269	2	0.003906
858	1.0.0.1	188.44.42.233	DNS	34185	12	0.007244
923	1.0.0.1	188.44.42.233	DNS	34883	3	0.017001
939	1.0.0.1	188.44.42.233	DNS	35644	1	0.000000
940	1.0.0.1	188.44.42.233	DNS	35645	2	0.004462
981	1.0.0.1	188.44.42.233	DNS	40838	1	0.000000
993	1.0.0.1	188.44.42.233	DNS	51739	3	0.080147
1068	1.0.0.1	188.44.42.233	DNS	68637	1	0.000000
1088	1.0.0.1	188.44.42.233	DNS	75356	1	0.000000
1119	1.0.0.1	188.44.42.233	DNS	75985	2	0.000051
1136	1.0.0.1	188.44.42.233	DNS	76797	1	0.000000

	mean_packet_size	std_packet_size	min_packet_size	max_packet_size
319	126.000000	0.000000	126	126
380	104.500000	12.020815	96	113
477	115.000000	8.485281	109	121
502	612.000000	0.000000	612	612
503	162.500000	74.246212	110	215
567	612.000000	0.000000	612	612
599	152.000000	0.000000	152	152
695	147.000000	0.000000	147	147
711	146.000000	7.071068	141	151
763	161.500000	2.121320	160	163
858	361.250000	230.647754	102	784
923	132.333333	28.023799	101	155
939	175.000000	0.000000	175	175
940	155.500000	14.849242	145	166
981	127.000000	0.000000	127	127
993	159.333333	3.785939	155	162
1068	128.000000	0.000000	128	128
1088	128.000000	0.000000	128	128
1119	153.500000	0.707107	153	154
1136	132.000000	0.000000	132	132

Рис. 8: Сравнение обычных и аномальных потоков: распределение признаков

Из таблиц видно, что аномальные потоки чаще всего используют протокол DNS и имеют значительное отклонение в размерах пакетов и продолжительности по сравнению с обычными.

10 Определение оптимального числа эксцессов

Для повышения точности детектирования превышений порогов был реализован подбор оптимального количества эксцессов — наблюдаемых выбросов — с наименьшей ошибкой при аппроксимации интервалов между превышениями экспоненциальным распределением.

```
model = PoT(data)
count_excess = [20, 50, 100, 200, 500, 1000, 2000, 5000, 10000, 50000]
for c_e in count_excess:
    model.fit(n_thresholds = 500, count_excess=c_e, yscale=False)
```

Рис. 9: обучение PoT для разного допустимого значения эксцессов

11 Анализ интервалов между превышениями

Временные интервалы между событиями превышения порога были рассчитаны и протестированы на соответствие экспоненциальному распределению. Это позволило формализовать модель появления редких экстремальных событий во времени. Ошибка при подгонке разностей моментов считается с помощью SSE (Sum of Squared Errors).

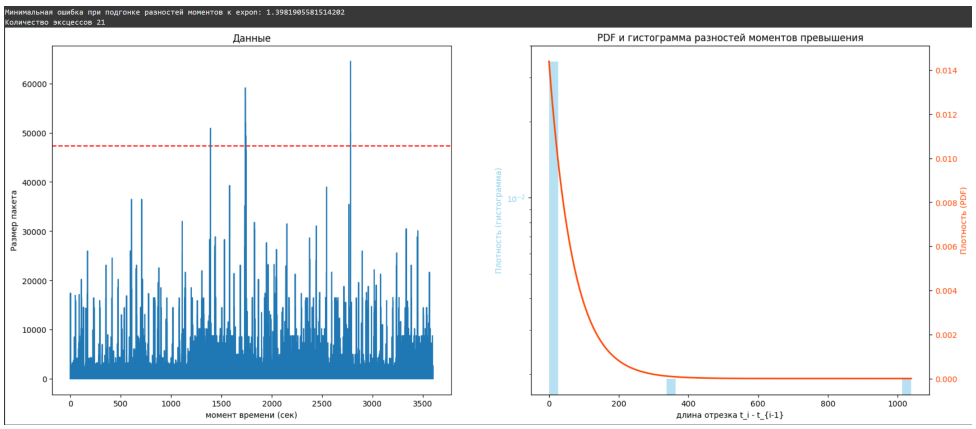

Рис. 10: Минимальная ошибка при подгонке разностей моментов к expon: 1.3981905581514202. Количество эксцессов 21

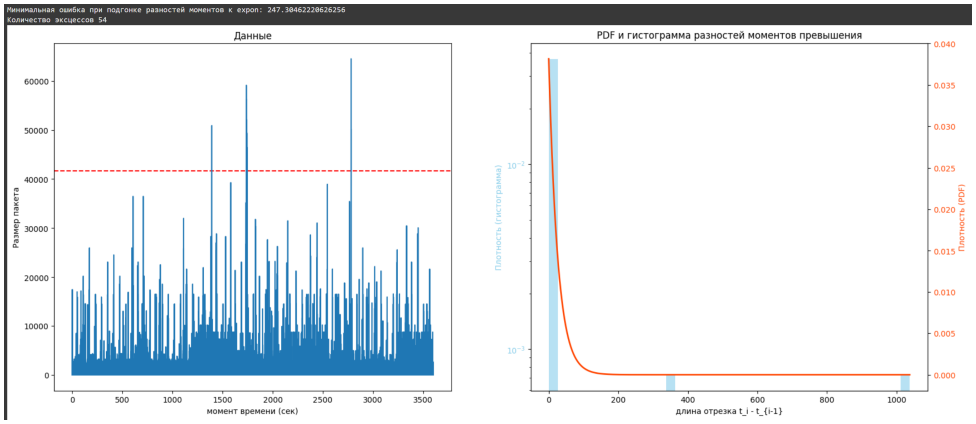

Рис. 11: Минимальная ошибка при подгонке разностей моментов к expon: 247.30462220626256. Количество эксцессов 54

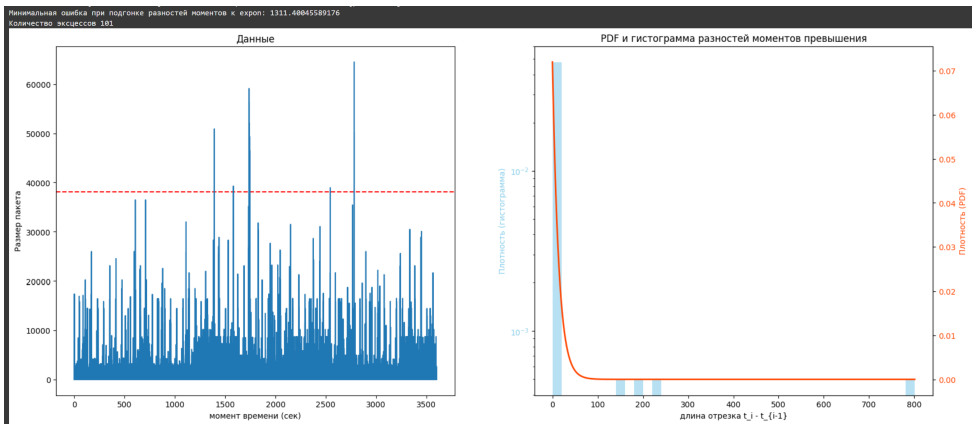

Рис. 12: Минимальная ошибка при подгонке разностей моментов к expon: 1311.40045589176. Количество эксцессов 101

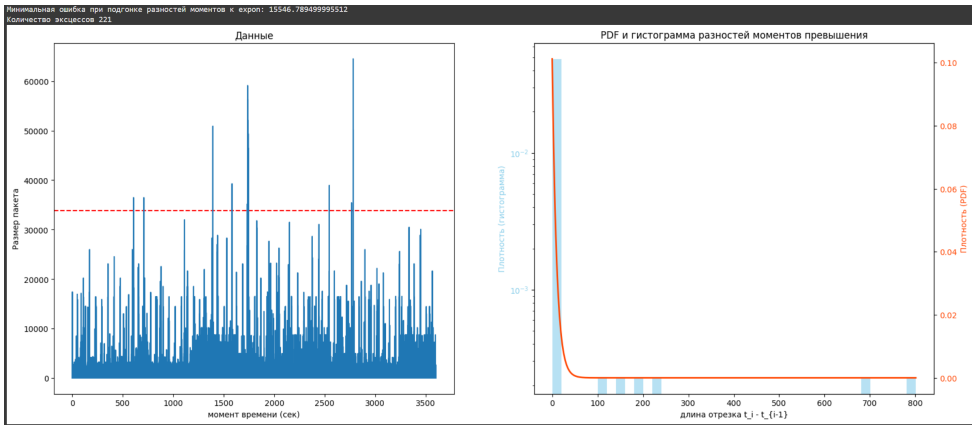

Рис. 13: Минимальная ошибка при подгонке разностей моментов к expon: 15546.789499995512. Количество эксцессов 221

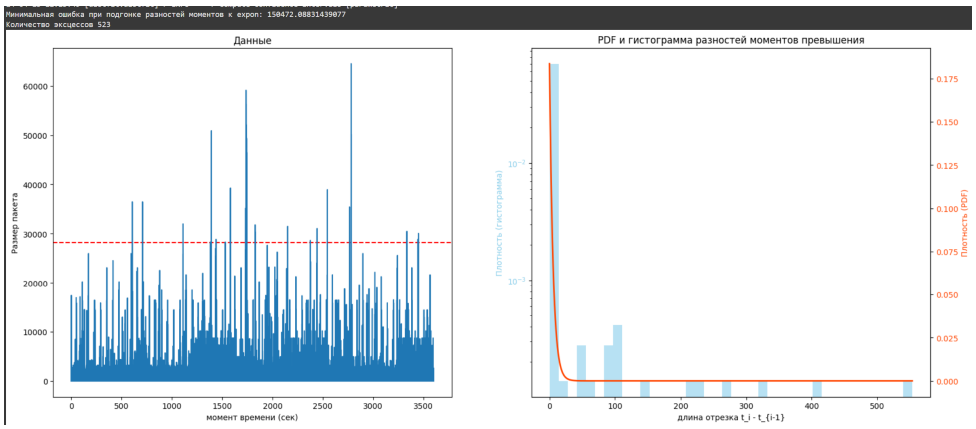

Рис. 14: Минимальная ошибка при подгонке разностей моментов к expon: 150472.08831439077. Количество эксцессов 523

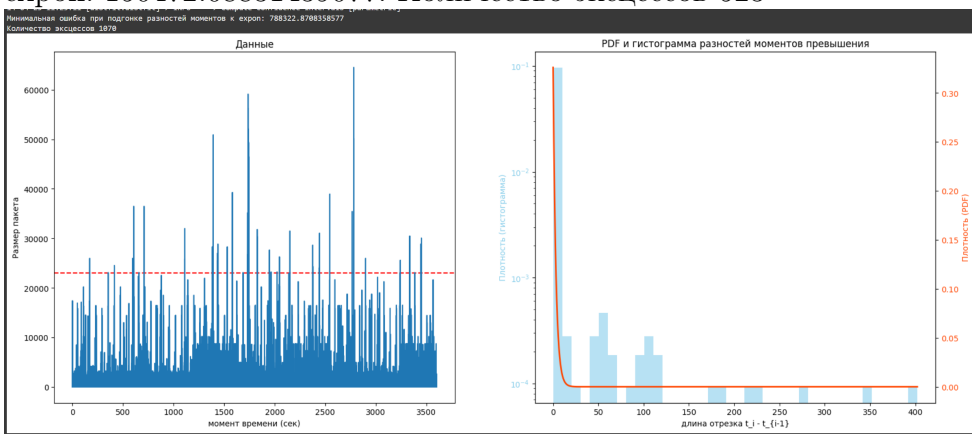

Рис. 15: Минимальная ошибка при подгонке разностей моментов к expon: 788322.8708358577. Количество эксцессов 1070

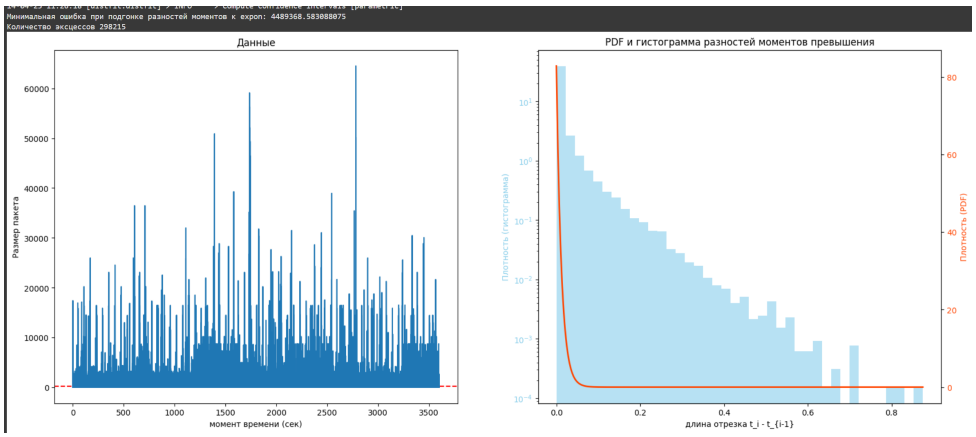

Рис. 16: Минимальная ошибка при подгонке разностей моментов к expon: 4489368.583088075. Количество эксцессов 298215

12 Вариационный автоэнкодер с нормализующими потоками (VAE + NF)

Для повышения гибкости аппроксимации латентного пространства вариационного автоэнкодера (VAE) была реализована модель с нормализующими потоками (Normalizing Flows). Такой подход позволяет заменить стандартное гауссовское аппроксимационное распределение $q(z|x)$ на более сложное, трансформируя базовую латентную переменную z_0 через последовательность обратимых функций:

$$z_K = f_K \circ f_{K-1} \circ \ldots \circ f_1(z_0)$$

Каждая функция f_k представляет собой простой, но дифференцируемый и обратимый оператор, в данной работе выбранный как **Planar Flow**. Его формула имеет вид:

$$z' = z + u \cdot \tanh(w^T z + b)$$

где $u, w \in \mathbb{R}^d$, $b \in \mathbb{R}$ — обучаемые параметры. Якобиан данной трансформации имеет простой аналитический вид, что позволяет вычислять логарифм детерминанта для корректного учёта плотности при обучении:

$$\log \left| \det \left(\frac{\partial f(z)}{\partial z} \right) \right|$$

Таким образом, модифицированный вариационный автоэнкодер максимизирует расширенный вариационный нижний предел (ELBO):

$$\mathcal{L}(x) = \mathbb{E}_{q(z_K|x)}[\log p(x|z_K)] - D_{\mathrm{KL}}(q(z_K|x)||p(z)) + \sum_{k=1}^{K} \log \left| \det \left(\frac{\partial f_k}{\partial z_{k-1}} \right) \right|$$

Применение к сетевым потокам

Модель была обучена на агрегированных сетевых потоках ‘flows$_d f$‘, :

- packet count — количество пакетов в потоке;
- duration — продолжительность потока;
- mean, std, min, max размера пакетов.

После стандартизации данных модель обучалась в течение 50 эпох. Для декодирования использовалась симметричная архитектура с одним скрытым слоем. В латентной переменной использовалось 10 измерений и 4 слоя нормализующих потоков.

Реконструкция потоков позволила вычислить ошибку восстановления (reconstruction error), на основе которой и проводилась детекция аномалий: потоки с ошибкой выше 99-го процентиля считались аномальными.

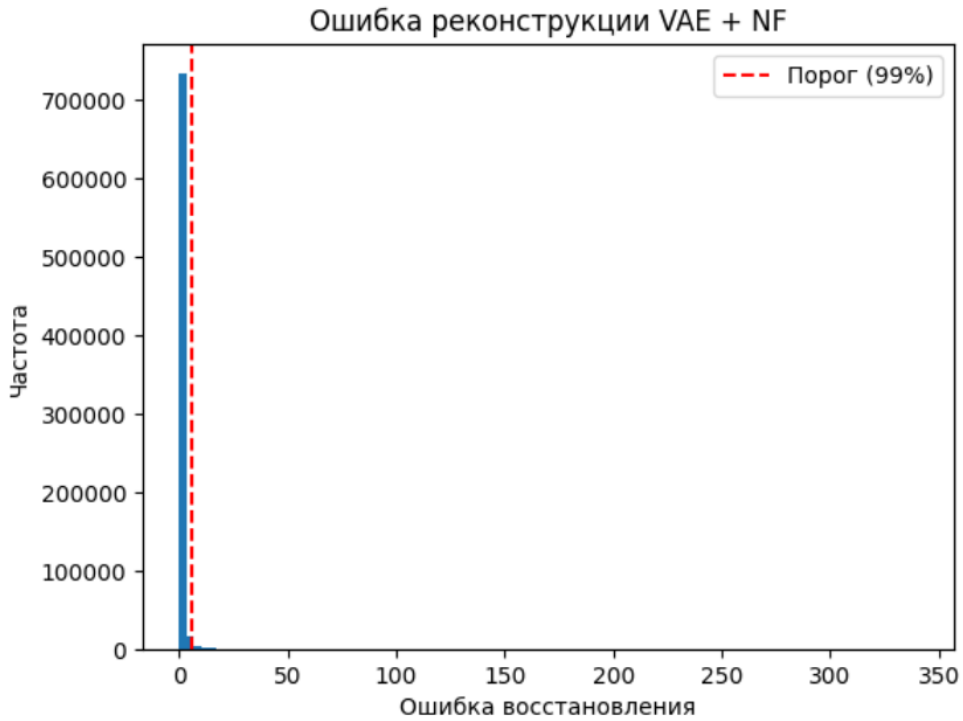

Рис. 17: Гистограмма ошибок реконструкции VAE+NF, красной линией — 99-й перцентиль

13 Выводы и обсуждение результатов

В ходе работы была проведена аналитика сетевого трафика, в рамках которой реализованы следующие этапы:

- Загрузка и исследование исходных сетевых данных;
- Преобразование пакетов в агрегированные потоки;
- Обучение вариационного автоэнкодера (VAE) и кластеризация в латентном пространстве;
- Выявление аномалий по ошибке реконструкции;
- Применение методов анализа экстремальных значений: POT;
- Формализация событийной модели на основе интервалов между эксцессами.
- Обучение модели VAE + NF

14 Заключение

В ходе данной работы данные были разделены на несколько групп с использованием латентного пространства, полученного из вариационного автоэнкодера (VAE), и последующей кластеризации методом KMeans. Наименее многочисленный кластер, как правило, может свидетельствовать о наличии аномальных наблюдений в выборке.

Также к исходным данным был применён метод Peak Over Threshold (POT), основанный на теории экстремальных значений. Однако, анализ графиков и числовых характеристик показал, что интервалы между моментами превышения порога не поддаются надёжному аппроксимированию экспоненциальным распределением, что, возможно говорит о неприменимости модели к исходным данным.

Дополнительно была реализована модель VAE с нормализующими потоками (VAE + NF), как предложено в статье [2]. Аномальными считались те потоки, чья ошибка реконструкции превышала 99-й процентиль.

Список литературы

[1] А. К. Горшенин, В. Ю. Королёв. *Определение экстремальности объемов осадков на основе метода превышения порогового значения*. Информатика и её применения, 2018, том 12, №4, стр. 16–24.

[2] Kingma, D. P., Rezende, D. J., Mohamed, S., Welling, M. *Variational Inference with Normalizing Flows*. Proceedings of the 32nd International Conference on Machine Learning, 2015.

[3] Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D. *Unsupervised Anomaly Detection via Variational Auto-Encoder for Seasonal KPIs in Web Applications*. Proceedings of the 2018 World Wide Web Conference (WWW '18), pp. 187–196.
DOI: 10.1145/3178876.3185996

[4] Adnan. *Training a Variational Autoencoder for Anomaly Detection Using TensorFlow*. Analytics Vidhya, 2023.
URL: https://www.analyticsvidhya.com/blog/2023/09/variational-autoencode-for-anomal